Penrose against Computational Theories of Mind

A Research Thesis

Presented in partial fulfillment of the requirements for graduation with research distinction in Philosophy in the College of Arts and Sciences of The Ohio State University

by

Kevin Gibbons

Project Advisor: Stewart Shapiro, Department of Philosophy

The Ohio State University
December 2015

Contents

Penrose against Computational Theories of Mind

Kevin Gibbons

December 18, 2015

Abstract

In 1961, J.R Lucas published an argument which called on Gödel's incompleteness theorem to show that the mind could not be simulated by a Turing machine. Though widely criticized, his sentiment found sympathy with mathematical physicist Roger Penrose, who published a new Gödelian argument in his 1994 book "Shadows of the Mind". However the idealizations necessary to get these arguments off the ground are substantial. Because of this, the scope of their conclusion is limited. In this paper, I'll appeal to these limitations to argue that there can be robust and useful versions of the computational theory of mind which hold up even in the face of the Gödelian arguments of Lucas and Penrose. More specifically, I contend that the arguments of Lucas and Penrose do not disqualify any computational theory of mind that is not already ruled out by the much less controversial results of Gödel and Benacerraf.

1 Background

Since its conception in the seminal work of Alan Turing, the formal notion of computability has been closely connected with the reasoning abilities of actual people. Indeed Turing makes this connection explicit saying: "We may compare a man in the process of computing a real number to a machine which is only capable of a finite number of conditions" [25]. It comes as no surprise then that computation was eventually turned back towards a rigorous treatment of traditional questions about the mind. These methods, pioneered by McCulloch and Pitts as early as 1943[18], attempt to explain the mind as being a complicated computing machine with its own unique set of formalisms. Such attempts proved to be remarkably effective, resulting in a long period of popularity for computationalist theories. And as more and more powerful computers are developed, these theories which treat the mind as a computing machine are only becoming more plausible.

However, these theories are not without their critics. One particularly interesting line of criticism appeals to Gödel's first incompleteness theorem. Gödel himself noticed this application of his theorem and presented it as follows in his 1951 Gibbs Lecture: "Either... the human mind (even in the realm of pure mathematics) infinitely surpasses the powers of any finite machine, or else there exist absolutely unsolvable diophantine equations"[10].

Gödel uses diophantine equations to drive his point home to an audience of mathematicians, but the general conclusion is this: Either the mind's ability to prove true mathematical propositions exceeds machine's ability to do so, or we can never prove every true proposition of mathematics. But the second disjunct doesn't sit well with our normal mathematical optimism. After all, experience has shown that if we just keep at it long enough, eventually we can decide even the most difficult problems (e.g. Fermat's Last Theorem)[1]. We see then that Gödel's conclusion is a measured one. All it shows with certainty is that this interesting disjunction holds. His reasons for thinking we should endorse one disjunct over the other are meta-mathematical. Mathematics proper, Gödel contends, has nothing more to say on the matter.

But others have argued that the incompleteness theorems make a stronger case against computationalism. Chief among them is J.R. Lucas. Lucas argues in his 1961 paper "Minds, Machines, and Gödel"[15] that the incompleteness theorems show with mathematical certainty that our minds are more powerful than machines. His argument (which I'll reconstruct in the next section) evoked a huge response in the literature. And while the vast majority of these respondents were critical of Lucas' argument, mathematical physicist Roger Penrose found it compelling enough to fashion his own attempt at a Gödelian argument. His first attempt, published in his 1989 book "The Emperor's New Mind"[21], is very similar to Lucas' original argument and thus contains the same defects. But his new argument, given in his 1994 book "Shadows of the Mind"[20], represents a significant departure from the logical structure of Lucas' original. This is why the new argument has proven to be much harder to refute. In fact, as I'll clarify later in this paper, Penrose's new argument puts us in a puzzling dilemma. The dilemma stems from Penrose's assumption of the soundness of normal mathematical practice. If this assumption is granted then, due to the logical properties of soundness, his argument goes through almost immediately. So Penrose's critics have taken issue with this assumption. But now things get tricky. On the one hand we are trying to defend computationalist approaches to cognition because they offer a powerful tool for understanding cognition. But in the course of this defense we are forced to appeal to an unseemly fact about soundness; namely, that whenever you assume a soundness scheme in the object language, absurdity immediately follows. It is the project of this paper to construct a defense that avoids this complication. To do so I'll look at the assumptions involved in these Gödelian arguments, and argue that they divorce Penrose's argument from most plausible computationalist accounts of cognition. So now we should look at these assumptions.

[1] Or, if we cannot decide them within a certain framework, at least we can understand clearly why not and in what framework we could, as was the case with Hilbert's Entscheidungsproblem

2 The Gödelian Dialectic

2.1 Idealizing Assumptions

The way these Gödelian arguments proceed is to consider the set of all arithmetic sentences that a particular person asserts which we can call **K**, following Shapiro[2][24]. They then assume that there is a Turing machine that would output all the members of **K**. Craig's theorem [6] says that any recursively enumberable set of sentences can be recursively axiomatized, so we can associate with this Turing machine a recursively axiomatized system F. By Gödel's theorem, if F is consistent, then there is a sentence that is true in F but that F doesn't prove. Since we understand Gödel's theorem though, we could see that this sentence is true and thus it would be in **K**. So this contradicts the assumption that **K** is output by the Turing machine because there is a member of **K** that the machine doesn't output; namely, its own Gödel sentence.

What then are the set-theoretic characteristics of **K**? It must be finite because the mathematician whose output determines **K** can only do so many problems before he dies. But the Gödelian can't accept this because for any finite set there is a Turing machine that outputs all and only the members of that set and then halts. So nothing about **K** would go beyond what is mechanically computable. To avoid this issue, we idealize away from things like lifespan and a finite number of particles in the known universe, and call **K** the set of sentences our mathematician would output with unlimited time and resources. So **K** is countably infinite.

The other main stipulation is that all of the members of **K** have to be true. So in a sense we're assuming an underlying competence of our mathematician, and simply ignoring any slips of concentration/judgement that might happen along the way. This assumption, on the meta-level where this debate is happening, guarantees us the consistency of **K**. This is a crucial assumption for the Gödelian to make. If we consider the actual set of outputs of our mathematician, warts and all, the set will certainly be inconsistent. But then, even if we're not sure a machine could output all the members of that set, one could output all the sentences logically entailed by that set. Specifically, the set of all arithmetic sentences. So **K** has to be consistent, or else the Gödelian loses by default (kind of). Restricting **K** to only true sentences prevents this. We have now that **K** is a countably infinite set of theorems of arithmetic which represents the output of some very idealized mathematician. We will take it on faith that this set is well-defined and that we do in fact refer to a specific set when we write or say '**K**'. The next task is now to take a detailed look at the arguments Lucas and Penrose give.

[2]We could just as well take **K** to be the output of some community of mathematicians or even all mathematicians in general, but there is nothing to be gained by this and it complicates the idealizations. So for ease of presentation we'll leave **K** as some arbitrary individual's output.

2.2 Lucas

Having made explicit the idealizations needed to get this argument up and running, we can now rigorously reconstruct Lucas' argument as follows:

1. Assume for reductio that there is a Turing machine W that outputs all the members of **K**.
2. This Turing machine, as shown above, has a set of outputs equivalent to the set of propositions provable by some formal system F
3. By the first incompleteness theorem, there is an arithmetic Gödel sentence G_F that is true, but not provable in the system F.
4. A human mathematician can see G_F to be true because he understands Gödel's theorem.
5. By 3 and 4, $G_F \in \mathbf{K}$ but $G_F \notin$ F
6. So W $\neq \mathbf{K}$, contradicting 1.
7. By reductio, there is no Turing Machine that outputs all the members of **K**

In this way Lucas purports to have shown that no Turing machine can simulate human arithmetical practice, and so minds outstrip machines. But whether he actually shows this to be the case is doubtful, as evidenced by the large amount of criticism he received after the article was published. The canonical objection is due to Putnam (1960)[22]. Putnam points out that the first incompleteness theorem doesn't establish that G_F is true. Rather, it establishes the conditional Con(F) $\rightarrow G_F$ where Con(F) is the typical arithmetic statement of consistency for a system. But why should we think our mathematician knows Con(F) to be true? After all, F might be an incredibly complicated system, and so he would have no way of determining whether it was consistent or not. But if he doesn't know Con(F), then he can't conclude G_F and so 4 no longer holds true. Thus, the reductio fails and mechanism is revived according to Putnam. So in order to save his argument, Lucas needs to secure the truth of Con(F)

Lucas' first attempt[16] at doing so is to claim that Putnam misunderstands the dialectical nature of the original argument. To him this is not a specific argument but rather an argument schema which is to be adapted depending on the particular Turing Machine that is advanced. Thus we get Con(F) simply from the candidacy of machine W. If Con(F) is false, then the output of W is inconsistent. But no mechanist would advance an inconsistent machine as modeling **K**. So they are committed to asserting Con(F) as well. Because the mechanist has secured Con(F) for us through their assertion that W outputs **K**, the argument goes through as planned. Or that's what Lucas argues anyway. Further, because it is an effective matter to list all the possible Turing Machines, Lucas could be

his own interlocutor, listing a Turing machine and subsequently showing that he is not that particular machine. In fact, he could spend forever going about this Sisyphean task, proving $W_1 \neq \mathbf{K}$, $W_2 \neq \mathbf{K}$, $W_3 \neq \mathbf{K}$ and so forth. So in this way even if he concedes that his proposed argument is only a schema, it is in no way less powerful than the original.

This response has received its own fair share of criticism. David Lewis for one argues (1979) that it doesn't get Lucas out of trouble[13]. To demonstrate why not, he has us consider two Turing Machines. Let's call the first one W_M and the second W_{NM}. The first machine is taken to simulate Lucas' normal theorem proving activity, while the second is to mimic his activity when accused of being a particular machine with Gödel number n. From this Lewis poses a dilemma: When Lucas produces the Gödel sentence for the machine the mechanist says matches his output, which machine exactly is this the Gödel sentence of? Assume it is the Gödel sentence for W_{NM}. In this case the position is exactly that of the original. We have no reason to suspect that Lucas can verify this sentence to be true, and thus he has not outdone the machine. Assume that it is instead the Gödel sentence of W_M. However, this sentence may very well be in the output of W_{NM}, and thus Lucas again has not beaten the machine. This would seem to show that the "dialectical nature" of Lucas' argument fails to save it.

2.2.1 Response from Selectivity

Another tack that Lucas takes in showing that we must be consistent is to point out that we will not assert just any arithmetic sentence. However, a formal system that is inconsistent will prove any sentence, so a Turing Machine that instantiates such a system would have an indiscriminate output. But we are discriminate in which arithmetic sentences we're willing to assert, so we cannot be such a machine.

Though intuitively plausible, this response won't get Lucas out of trouble either. The issue is that he equivocates between the axioms of a formal system and the theory entailed by those axioms. To illustrate this consider the Turing machine that, when started on a blank piece of tape, writes the binary code for '0=1' and '$0 \neq 1$' and then halts. Clearly this machine is inconsistent. But whether it "asserts" every sentence of arithmetic depends on how we define its output. If we take its output as just those sentences which it writes down before halting, then there are an infinite number of sentences it didn't assert. So it is just as selective as Lucas, or anyone else for that matter. If we take its output instead to be the set of sentences entailed by '0=1' and '$0 \neq 1$' under the typical rules of inference for first order logic, then Lucas is right; its output is just the set of all arithmetic sentences. Under this definition of output we have no reason to believe Lucas is selective in which sentences he asserts either. Unless he has never asserted two contradictory arithmetic statements, under this broad understanding of output Lucas actually outputs all sentences too. So to show us he was indeed more selective than the machine, he would have to show that he is consistent. If he could do that, he would win regardless because no machine is capable of showing its own consistency. We have no reason to think he can, though. So he is back to

where he started, with nothing to show for worrying about selectivity.

2.2.2 Gentzen and Transfinite Consistency Proofs

One interesting response Lucas has given appeals to Gerhard Gentzen's 1938 consistency proof for arithmetic[8]. In response to the charge that we can't establish Con(F) and thus don't know G_F, Lucas cites this consistency proof, arguing that just because we can't prove Con(F) via arithmetic methods, doesn't rule out our having mathematical ways of doing so. In particular, Gentzen's. But once we've established Con(F), we get G_F and so his anti-mechanistic proof goes through.

This seems like an ill-advised tactic though, given the nature of Gentzen's proof. The proof itself relies on a reduction method for derivations of a contradiction (or in Gentzenian terms, sequents ending in the empty set) and an induction principle on transfinite ordinals up to ϵ_0. The reduction method postulates that there is a sequent of arbitrary complexity ending in the empty set. Then it shows how this if that sequent exists, then a less complex one exists, and so on and so on. The procedure terminates in a minimally complex sequent ending in the empty set. But it is shown that this sequent cannot be derived, so there cannot be an arbitrarily complex sequent ending in the empty set. But if arithmetic were inconsistent there would have to be. So arithmetic is consistent.

While at first glance this might seem like just the proof Lucas needed, it doesn't actually gain him any ground. Because although the methods in this proof are not finitary, as von Plato points out in his SEP article on the subject (2014) they are completely constructive. And as such, they should be accessible to a Turing machine. So again, Lucas has been matched by a new machine which he needs to show to be consistent. But Gentzen's subsequent proof established that the consistency of the induction principle + PA is not provable in ordinary arithmetic [9], so Lucas seems unable to do so. Since he can't know the new machine to be consistent, he cannot know its Gödel sentence to be true, and so his argument fails again.

2.3 Penrose's New Argument

So despite a valiant defense, Lucas' anti-mechanistic arguments appear to be beyond repair. Now let's look at the argument Roger Penrose gives in Chapter 3 of "Shadows of the Mind". The argument is summarized in a dialogue between a programmer (Albert Imperator aka AI) and a super computer he designed. The computer tells Albert that the theorems it have proven are all "unassailably" true, and Albert tells the machine that it can be considered equivalent to a Turing-Machine M. The computer protests, saying that if Albert could know both of these were true, then there would be a theorem that Albert could prove that it couldn't- a possibility its robotic pride won't admit. In response Albert has the robot consider the theory T(M) defined to be the set of all sentences which follow "unassailably" from those outputted by M and the assumption that it is equivalent to M. The robot then

concludes by the following argument that it cannot in fact be equivalent to M:

1. Assume that I am equivalent to the Turing machine M.
2. Because all the sentences I output are unassailably known, all the sentences I output are true.
3. By (1) and (2), all the sentences output by M are true.
4. Because all the sentences output by M are true, all the sentences in T(M) are true.
5. Because M is recursively enumerable, T(M) is effectively generated, and so itself recursively enumerable.
6. So T(M) is equivalent to some Turing Machine M', instantiating some formal system F'.
7. By Gödel's theorem, F' has a Gödel sentence such that $G_{F'}$ is equivalent to Con(F') and $G_{F'} \notin$ F' .
8. By (4) all members of T(M) are true, so F' is consistent
9. So $G_{F'}$ is true.
10. Discharging (1), I conclude if I am equivalent to M, then $G_{F'}$ is true.
11. By definition then $G_{F'} \in$ T(M) and so $G_{F'} \in$ F'. But this contradicts (7).
12. So I am not equivalent to M, by reductio.[3]

2.3.1 Criticism of the New Argument-Assumption of Soundness

The most damning critiques of this new argument focus on its assumption of soundness in (2). The standard objection along these lines is raised by Chalmers and Shapiro [4][24]. They object that once we have introduced a soundness scheme that is accessible in the object language, it's a simple thing to derive a contradiction. To highlight this, let's define the predicate K($\ulcorner\phi\urcorner$) so that K($\ulcorner\phi\urcorner$) $\iff$ ϕ is "unassailably known". Let's assume further that K(x) fulfills the standard requirements of a Hilbert- Bernays provability predicate for PA (because if our robot knows a theorem of arithmetic unassailably, then it only makes sense it has found a proof for it in PA or some weaker system). So explicitly:

1. If PA $\vdash \phi$, then PA $\vdash$ K($\ulcorner\phi\urcorner$)

[3]It should be noted that while Penrose takes this argument to be a refutation of the hypothesis that we are actually machines, it does just as much work against the idea that our arithmetic practice can be simulated by machines. Simply change statements like "I am equivalent to M" to "My arithmetic output is equivalent to that of M", and the argument goes through just the same against this weaker hypothesis.

2. PA $\vdash$ K($\ulcorner \phi \rightarrow \psi \urcorner$) $\rightarrow$ (K($\ulcorner \phi \urcorner$) $\rightarrow$ K($\ulcorner \psi \urcorner$))

3. PA $\vdash$ K($\ulcorner \phi \urcorner$) $\rightarrow$ K($\ulcorner$K($\ulcorner \phi \urcorner$)$\urcorner$)

From here then we can derive a material implication version of Löbs theorem for K(x) of the form: K($\neg$ K($\ulcorner \phi \urcorner$) $\vee$ ϕ) $\rightarrow$ K($\ulcorner \phi \urcorner$). But when we introduce the soundness scheme K($\ulcorner \phi \urcorner$) $\rightarrow$ ϕ then we run into trouble. For any ϕ either K($\ulcorner \phi \urcorner$) or $\neg$ K($\ulcorner \phi \urcorner$). In the former case, the soundness scheme gives us ϕ outright. In the latter case $\neg$ K($\ulcorner \phi \urcorner$) gives us itself. But in both cases, the antecedent in Löb's theorem has been fulfilled and so we can conclude K($\ulcorner \phi \urcorner$) for all ϕ. One more application of the soundness scheme tells us that any sentence ϕ *whatsoever* is true! This is clearly absurd, so there is something wrong with assuming the soundness of K.

This is very much along the lines of what Chalmers concludes. Shapiro notes this fact, but then proceeds to the more fundamental worry that the statement itself of the soundness schema is problematic. Both of these are good things to worry about, but they seem troubling to our normal mathematical intuition. A mathematician doesn't only think the methods he is using are sound when he goes home and does meta-logic regarding what he accomplished during the day. He must think that his methods are sound when he is actually using them. Otherwise what reason could he give, in that moment, for using them as opposed to some other? This is an unresolved issue for logicians, but still, a defense of the computational theory of mind that avoids it would be preferable. So it is at least worthwhile to look at some of the other responses to Penrose's new argument which don't involve questions about soundness.[4]

2.3.2 Per Lindström

One such objection comes from Per Lindström [14], and goes as follows.

1. Let M be the Turing-Machine which supposedly models my $\prod_1$ arithmetic output.

2. Let the output of M be equivalent to some set F of $\prod_1$ sentences plus the sentence $\neg$Con(F).

3. By Gödel's second incompleteness theorem this set is consistent.

4. Under the assumption that I am sound, M is sound.

5. If M is sound, then M is consistent.

6. So Con(M) $\in$ T(M).

[4]Interestingly, Penrose's assumption of soundness gets his own theory of mind into trouble because the quantum computing Penrose postulates accounts for consciousness has built into it an infinitesimal chance of error. So as the number of outputs of our idealized human quantum computer goes to ∞, the probability of error goes to 1. So the idealized mathematicians' output cannot be sound.

7. But since F $\subset$ M, Con(M) implies Con(F).

8. Since T(M) is closed under deduction, Con(F) $\in$ T(M).

9. But since M $\subset$ T(M), $\neg$Con(F) $\in$ T(M).

10. So Con(F) and $\neg$Con(F) are members of T(M) and therefore T(M) is inconsistent.

11. Therefore T(M) is not sound.

But 11* acts as a counter example to the premise (3) of Penrose's argument, so it doesn't go through.

Although Lindström's tactic here appears to be to show that one of Penrose's premises is not always true, it is really just a soundness concern in disguise. Notice that the assumption 5* features importantly in Lindström's argument. The disguise then is that he contests a later premise in Penrose's argument, but is still using the assumption of soundness to derive a contradiction in the course of that contesting. So Lindström's objection, while subtle, isn't what we're looking for.

2.3.3 Bringsjord and Arkoudas

Another objection to Penrose's new argument is due to Bringsjord and Arkoudas [3]. They claim that Penrose conflates the object language with the metalanguage and that once we set those straight, there is no contradiction at all. Specifically, they claim "(1) G(F) is true on the one hand, and yet (2), which says that F cannot conclude G(F), is true on the other. But wait a minute; look closer here. Where is the contradiction,exactly? There is no contradiction. The reason is that (1) is a meta-mathematical assertion; it is a claim about satisfaction". This is an interesting claim, and would be a serious charge against Penrose were it true.

Fortunately for Penrose, it's not. In fact, Bringsjord and Arkoudas seem to have missed the subtlety of Penrose's new argument completely. First of all, instead of characterizing T(M) as the set of all sentences that unassailably follow from M's output and the hypothesis that "I am M", they describe is as the set-theoretic union of these two. This is not what Penrose meant though, and the distinction is important for his argument. This might explain why they claim Penrose finds the fact that G(F') is true and the fact that F' doesn't decide G(F') to be contradictory, which of course he doesn't. So while they are certainly right to point out that there is some reckless back and forth between the object language and the meta-language, their claim that this undermines Penrose's argument on its own is rather unconvincing.

3 A new criticism of the New Argument

So now the dialectic sits at an interesting juncture. The most effective counterarguments against Penrose show his assumption of soundness to lead to logical contradiction. But in doing so they run headlong into a fundamental problem in the foundations of mathematics. This seems unideal. We want to be able to defend computational accounts from Penrose without diving into a host of foundational debates. For the rest of the paper I'll attempt to give such a defense. The tactic will be to grant Penrose that the new argument is valid, but that the assumptions it needs to make to invoke Gödel's theorem are not assumptions involved in some computational theories of mind. As such, they offer the mechanist a good place to resist Penrose. The main idealization I'll worry about are 1) That humans are sound[5] and 2) that CTM gives the actual algorithm to model human output. I'll then show how these assumptions Penrose takes as granted entail epistemic commitments for CTM that no one in the field would agree to. But, if they did agree to them, then they would be susceptible to the arguments of Gödel and Benacerraf, making Penrose's stronger argument superfluous. These considerations together will show that even if Penrose's argument were valid, it would be surprisingly impotent.

3.1 Taxonomy of Theories

Up until now, I have followed the style of the logical literature in treating computational theories as a coherent whole. This treatment is too simplistic though, as there are a number of distinct sub-theories in computational psychology with their own unique assumptions and methodologies. While the goal of this paper is not to give a comprehensive account of all these different theories, it will be necessary going forward to lay out some basic categories which these theories fall into. The tack will then be to examine how the assumptions made in Penrose's argument feature in each of the categories. In laying out and describing the categories, I'll follow the lead of Margaret Boden [2], who analyzes how the notion of "computation" differs among computational theories. These differences let us distinguish between *formalist* computational theories and *competence* computational theories. As it will turn out, formalist theories are not subject to Gödelian arguments whereas competence ones are. But in order to see why that is, we need to look at how the two types differ.

3.1.1 Formalist computational theories

The first definition of computation we'll consider is the one that features in formalist theories. These theories define 'computation', as Boden writes, "as the formal manipulation of abstract symbols, by the application of formal rules"(pg.229). So this definition matches

[5]This is different from the objections of Shapiro and Chalmers because while they are concerned with how soundness leads to logical contradiction, I'm concerned with how they divorce the argument from many CTM.

closely the definition of computation that features in computer science, formal logic, etc.[6] This is a boon to the formalist theories then because they are able to bring all the results and methods of these other fields to bear on questions about the mind.

Exactly what role these formalisms are playing when they are employed in a computational theory is a matter of some controversy though. On the one hand, some theorists consider the notion of computation as a useful explanatory metaphor. Whether or not the brain is actually a computer is a question that may be beyond our powers of explanation at the moment, but it has allowed to precisify a number of questions about how cognition works. So on this view, as long as the assumption that the brain is some kind of computer keeps yielding results, that is all the evidence needed to belive provisionally that the assumption is true. On the other hand, some theorists make the stronger claim that the brain literally does computations in exactly the same way that a computer does. One such theorist is Zenon Pylyshyn, who draws on the notion of strong equivalence from computer science to clarify this claim[23]. For him, mental processes are physically instantiated in the brain in the exact same way as programs are instantiated in a computer's functional architecture. So then when we model these mental processes computationally, we are not speaking metaphorically, but rather we're describing what is actually going on in the mind. This is an important distinction to highlight, because the formalist theories which use computation metaphorically will exempt themselves from Penrose's argument more easily than those which use it as a modeling tool.

3.1.2 Competence computational theories

The notion of computation outlined above is the definition used in most of computational psychology. As such, if any theory utilizing this notion can exempt itself from Gödelian arguments, then the majority of the field is exempted. By way of contrast, take the second notion of computation that Boden highlights, due to David Marr[17]. In his work, Marr distinguishes between three different levels of explanation that a sufficient account of mental processes must contain: Computational, algorithmic, and mechanical. The mechanical level is concerned with how the algorithmic level is implemented physiologically, and the algorithmic level matches closely the types of explanation that formalist theories intend to give. The level unique to Marr then is what he refers to as the "computational" level. What Marr refers to when he talks about 'computation' is a strict input-output description of some information processing task, abstracted away from the formalism that carries it out. To illustrate this point, Marr has us consider the "computational theory" of a cash register at a grocery store. He argues that if the register is to perform its job correctly, then it necessarily must satisfy the following properties:

[6]The scope of this definition is a little uncertain. In particular, it's not at all clear whether connectionist models of information processing can be captured under this formulation of 'computation'. For our purposes though, it is enough to worry about those theories that claim that the computations the mind does are Turing computable. Which theories in fact turn out to do so is another question.

1. If you buy nothing, you pay nothing.

2. The order you buy things doesn't matter.

3. Buying one batch of things first, then another batch of things next is the same as buying them altogether.

4. If you buy something, and then return it, you have the same amount of money you had before.

Since these constraints are really just the axioms for addition and zero in Robinson arithmetic, Marr has us conclude that arithmetic is the proper computation for the register to use. So then the computational theory does two things. First, it makes *a priori* arguments for constraints on the operation, and secondly it gives an account of what operation satisfies these constraints. In short, the computational level gives the "what" and the "why", whereas the algorithmic and mechanical levels give the "how".

Marr's notion of 'computation' is striking for a couple of reasons. First, Marr refers to as 'computational' is more of a theoretical task-analysis than a traditional computation. By saying that there is a level of computation that is independent of how it is actually implemented, he is deviating from the accepted notion of computability. This separation is not completely unique to Marr, as a comparable distinction between "competence" and "performance" appears in linguistics due to Chomsky. For Marr, an account of a computation isn't concerned with how we actually carry it out (or whether it is possible to do so), but rather what we would have to do if we were to carry it out. For this reason, Boden refers to theories that take after Marr as "competence" theories. Secondly, Marr thinks that we can give a substantive account of the computational level of a theory by analyzing *a priori* constraints on the task being carried out. But now notice that these assumptions that Marr is making run parallel to those the Gödelian makes when idealizing about the set **K**. In the next subsection, we'll show why this poses foundational issues for any theory that takes after Marr.

3.2 Penrose vs. CTM, assumption wise

3.2.1 Epistemic foundations: Metaphor vs. Model

So now we are in a position to compare the assumptions made by the Gödelian to those that are made by different computational theories. Beginning with the formalist theories described above, it is immediately apparent that those which consider the use of computational notions to be merely an important metaphor are not susceptible to the criticisms of Lucas and Penrose. After all, both Lucas and Penrose argue by way of contradiction. But if the assumption for contradiction (i.e. (1)- My arithmetic output is equivalent to that of some Turing machine) is not something that the theorist takes to be literally true, then it's no issue when it's shown to lead to absurdity. It doesn't matter if an assumption

that doesn't feature in their research program is contradictory; as long as it keeps yielding results, using computational methods is unproblematic.

As has already been said, not all who use this formalist notion of computability in their research take it to be metaphorical. There are researchers in computational psychology (e.g. Pylyshyn, Newell, Simon, etc.) who take the assertion that the mind is doing computations to be literally true. But then it would seem like Penrose is off to the races because he is now justified in assuming (1) and so his reductio argument, assuming it's otherwise valid, ought to go through. Since these theorists cannot extract themselves as easily as those working with a metaphorical understanding, if they want to resist Penrose, they need to find another assumption to take issue with.

The best next move for the computationalist is to object to Penrose's assumption of soundness. Following Putnam's original objection to Lucas, the computationalist may question what grounds Penrose has for asserting that our arithmetic output is sound. But the computational theorist does not have the same liberty that Putnam has to make this objection. Because while Putnam comes to the debate with no backgrounds commitments, the computationalist has a horse in the race. They have ostensibly given an account in the manner of computational psychology of our theorem proving activities. It would be a truly remarkable feat if they managed to so do without appealing to any facts of arithmetic. But barring the extraordinary, their arguments for their theory of arithmetic theorem proving will itself rely on certain theorems of arithmetic. So to deny soundness would at first blush seem to undercut their own work.[7]

This is not to say though that the formalist has no outs. It just means that some subtlety is required. The subtlety involves distinguishing between the macro-process of theorem proving generally, and the different sub-processes that make up the larger scale process. For example, when we count (i.e. sum a series of 1's) there is evidence that we employ some sort of language processing, whereas we judge inequalities using visuo-spatial representations [7]. So although both proving things about inequalities and proving things about sums are part of the general process of "doing arithmetic", they themselves are very different processes and would have different computational representations. But then this gives the formalist his out. Because while it seems very likely that some arithmetic reasoning would have to be involved in the defense of their computational theory, there's no reason to think this defense would exhaust all the different processes that make up arithmetic reasoning. If he is able to isolate those processes involved in his theory and give reasons (not necessarily *a priori* ones!) why these processes are internally sound, then he has adequately defended his theory. But he makes no commitments to the global soundness of our arithmetic practice, and thus doesn't have to grant Penrose his assumption. Thus, Penrose's argument does not undermine his computational theory.

Before looking at how Penrose's argument fairs against competence computational the-

[7]They could of course just remain neutral on the topic of soundness, and avoid the issue in this way. But we'll consider what happens if they want to give a stronger defense of their theory.

ories, I want to head off a possible misunderstanding about the formalist's exemption from Penrose. The misunderstanding arises from this line of thinking: if the formalist can show that the methods they used to derive their theory are arithmetically sound, then how is it that the theory itself could be unsound? While this is an intuitive response, it doesn't correctly account for the nature of the formalists work. Even though there is arithmetic reasoning involved in giving a computational account of arithmetic practice, there is also wealth of empirical evidence (e.g. psychophysiological data from fMRIs) that goes into the construction of computational theories. So then the formalist can both contend that he made no errors in any sort of arithmetic used in interpreting the data, constructing a model, etc. while also leaving open the possibility that various sections of the model may contradict each other, making the model itself inconsistent and therefore unsound. But such general questions about soundness lie outside the scope of a formalist theory. Interestingly, Marr's introduction of his higher "computational level" brings these questions about global soundness into the fold of his theory, and thus get him into trouble. We can now proceed to see why exactly this is the case.

3.2.2 Penrose's due: Competence theories and soundness

So we saw above that Penrose's argument fails to do work against formalist computational theories because his assumption of soundness isn't something that the formalist is committed to. But on the other hand, competence computational theories make more ambitious foundational assumptions, and thus are vulnerable to Penrose's argument.[8] To show this we have to establish two things: First, that the "computational level" of the competence theory picks out a set of arithmetic sentences with the same properties of **K** described in 2.1 (i.e. recursively enumerable and countably infinite). Second, that the competence theorist cannot avoid a commitment to soundness in the same way that formalist can. If we can show these are the case, then it will be easy to show that Penrose's argument goes through against competence theories.

That a competence theory of arithmetic picks out a set of arithmetic sentences that has the same properties of **K** follows directly from the definition of the "computational level" and from the distinction between competence for a computation and performance in a computation. Recall from above that "computational level" of a competence theory establishes constraints on a certain computational task that its inputs and outputs must satisfy if it is to do that task at all. Further, for the competence theorist, these constraints are derived *a priori* from the definition of the task being carried out. As Boden summarizes it, "What is done- and how it must be done if it is to be done at all- is the focus of interest [of the computational level]". But then given these facts, the constraints on a competence

[8] Or rather, they would be if there were any competence theories in computational psychology concerning how we do arithmetic. As far as I know, there are no such theories. However, given how influential Marr's work was in the field, it is not too far-fetched to think that one day someone may attempt to construct one. The goal here is to preempt any such attempt.

theory of arithmetic turn out to be simply the usual axioms for first order arithmetic. This is because the operations of arithmetic are defined by these axioms, so if you are doing arithmetic then you must be working within the scope of these axioms. In terms of input and output then, given an input of some set of arithmetic sentences, the proper output would be the set of arithmetic sentences which follow from the input set and the axioms of arithmetic (let's specifically let these axioms be those of **Q** for the sake of clarity). So any computation that can be said to be doing arithmetic has to output, given the same input, at least a subset of the larger set.

We have now a set of arithmetic sentences as the output of this competence theory, and so all that's left to establish is that its cardinality is countably infinite. To do so, we need to revisit the distinction between "competence" and "performance". As mentioned above, this distinction is originally due to Chomsky, and is probably best explained in it's original context. For Chomsky, the distinction arose with respect to how we make use of the rules of a generative grammar i.e. the rules of grammar that tell us how to construct sentences. While performance is concerned with the actual construction of sentences, Chomsky writes "[competence] is concerned primarily with an ideal speaker-listener, in a completely homogeneous speech-community, who knows its language perfectly and is unaffected by such grammatically irrelevant conditions as memory limitations, distractions, shifts of attention and interest, and errors (random or characteristic)"[5]. By following out this line of reasoning, Chomsky conlcudes that humans have the competence to generate an infinite number of sentences. So by now the parallel should be apparent. Instead of talking about the generative rules for a natural language like English, we are looking at the axioms of **Q**, and instead of considering those English sentences generated by an "ideal speaker-listener", we consider those arithmetic sentences asserted by an "ideal mathematician". So by utilizing this Chomskian notion of competence, the competence theorist has made the exact same idealizations that the Gödelian does. So, because the set of sentences which follow from the axioms of **Q** is effectively generated and has infinite cardinality, the competence theorist is committed to the fact that the set of arithmetic sentences that make up the output for his theory be countably infinite. Thus, at the "computational level", a competence theory picks out a recursively enumerable set **K** of arithmetic sentences.

At this point things are looking good for Penrose. The competence theorist is committed to the idealized output **K** of some human mathematician being recursively enumerable and therefore equivalent to the output of some Turing machine M. This gives Penrose his reductio assumption (1). If he is granted further that **K** is sound, then he can push his argument through. Now Penrose has to say, as he did to the formalist, that the assumption of soundness is implicit in the competence theorist's defense of his model of human arithmetical practice. The competence theorist can reply in one of a few ways. He can attempt to make the same counter-argument that the formalist made; namely, that his claim of soundness is a restricted one and thus doesn't give Penrose a strong enough assumption to carry his argument through. Should this response fail, the competence theorist could then claim that he uses some other reasoning besides arithmetic reasoning

to come to see that his theory is sound. Lastly, he might just deny that he has to assert his theory to be sound. It would clearly be ideal then if one of the first two lines worked. As we'll see though, because the competence theorist insists that a complete computational theory gives an account of the "computational level", they don't.

Let's first pursue the formalist's response, and see how it works for the competence theorist. When faced with the charge that they can't assert a computational theory of human arithmetical practice that isn't sound because the defense of the theory relies on arithmetic, the tack was to restrict the claim of soundness to only those involved in proving the theory. Thus they deny Penrose the global soundness needed to make his argument, and free themselves up to develop theories at the "algorithmic level" (to use Marr's term mentioned previously). But for the competence theorist, the introduction of the "computational level" complicates things. Because by asserting that the arithmetic the competence theorist used in contracting his model is sound, he asserts that the rules governing that arithmetic (i.e. the axioms at the computational level) are also sound. But then, so long as he utilizes addition and multiplication in defending his theory, he has implicitly asserted that the axioms generating **K** are sound. But now Penrose has the assumption for (2) in addition to that of (1), so he has all he needs to level his argument.

In response, the competence theorist could then deny that the reasoning used in seeing the soundness of his theory were arithmetic. However, it could be argued that this leads to regression similar to those seen in association with Turing-Feferman reflection principles. Because by claiming that he sees his competence theory to be sound, he implies that he sees it to be consistent. But the consistency of his system can be rendered as an arithmetic sentence. Thus, regardless of the method he used to come to see consistency, he has come to assert an arithmetic sentence. Thus, this ought to be captured in the the output set **K** [9]. Of course by Gödel's theorem, whatever mechanisms the competence theorist postulates at the algorithmic level cannot output **K** will not output Con(**K**). So in order to capture Con(**K**) he will need to postulate a new output set $\mathbf{K}_1$ of some augmented theory. But then the dialectic repeats, leaving us with a $\mathbf{K}_2$ and then a $\mathbf{K}_3$ and so on. However, unlike the trasnfinite numbers in T-F reflections, the human mind does not go on forever. So at last there would have to be some $\mathbf{K}_n$ such that given m $\leq$ n, $\mathbf{K}_m \subseteq \mathbf{K}_n$. Thus the arms race would end and Penrose could wage his argument using $\mathbf{K}_n$ in (1).

So the last option for the competence theorist would be to remain completely impartial to the soundness of his theory. Clearly, this option denies Penrose his assumption for (2) and thus his argument wouldn't go through. But notice that Penrose can still claim the following conditional: If the mind is in fact a computer, then no competence-style computational theory of human arithmetic practice can be completed and known to be sound. This seems to be a fairly damning result for the competence theorist.

[9] At this point, the competence theorist might object, because **K** is ostensibly the output an ideal mathematician, and how closely the theorist matches the idealized mathematician is not known. To make this objection though, the theorist would have to steer clear of Preface Paradox like concerns. Whether or not this is possible is an interesting question, but too involved to go into at the moment.

3.2.3 Giving Gödel and Benaceraff their dues too

While Penrose's conditional is a fairly striking result, it is not an altogether unique one. Consider the logically equivalent statement "It is not the case that the mind is a computer and also a competence-style computational theory of human arithmetic practice can be completed and known to be sound. But compare this to another quote from Gödel's Gibbs lecture: "However, if [a finite rule generating **K**] exists, then we with our human understanding could certainly never know it to be such...we could never know with mathematical certainty that all propositions it produces are correct"[10]. So then while there might be a system that generates all the truths of "subjective mathematics", it could never be known to be such and also be known to be consistent, which precludes it from being known to be sound. This is just a restatement of Penrose's conclusion then, but it is a restatement that is much more direct and came 43 years prior to the publication of SOM. So given this Penrose's argument seems an unnecessary complication.

Now it may be objected: "Yes, Gödel anticipated Penrose's result, but Penrose explicated the logic and made it more precise. So it wasn't a useless effort". This is true, Penrose gives a clever and exact argument whereas Gödel's talk is more along the lines of a proof sketch. But this doesn't save Penrose from the charge of redundancy, because in 1967 Paul Benacerraf weighed in on the matter with a very exact 20 step derivation[1]. At the end of it, he writes: "At best Gödel's theorems imply... that given any Turing machine W_j, either I cannot prove that W_j is adequate for arithmetic, or if I am a subset of W_j then I cannot prove that I can prove everything W_j can.... In a relevant sense, if I am a Turing machine, then perhaps I cannot ascertain which one". He adds as an afterthought, "Of course, I might be an inconsistent Turing machine." Obviously, if you are an inconsistent Turing machine, then it is possible to assert your own consistency, but that assertion just happens to be wrong. But then this result also captures the conditional Penrose has shown. Because if we are a computer, then either we can't determine what our own program is and so can't determine the axioms governing our output or we are an inconsistent computer. But if we can't determine the axioms or the program, then neither the computational level nor algorithmic level of a competence computational theory of arithmetic would be completable. So then we conclude that if we are a computer, either we are an inconsistent one, or we cannot give a complete competence computational theory of arithmetic, which is exactly what Penrose is entitled to conclude.

So now, finally, we have demonstrated everything involved in my "new criticism" of Penrose's argument. On the one hand, it was shown that Penrose's arguments fail to do significant work against formalist theories in computational psychology, which account for the majority of the field. On the other hand, the work it does against competence theories is not unique to it. The same work is done by the arguments of Gödel and Benacerraf, and these arguments are clearer and more precise. Therefore Penrose's argument is at best superfluous.

4 Conclusion

Having criticized Penrose's argument, all that is left to do before ending this discussion is to briefly examine where the debate stands. And while the arguments have done work to show exactly where Penrose's argument (were it valid) would stand in the dialectic, nothing in this paper, including Penrose's argument, has advanced the dialectic itself. So I'll end with a brief treatment of what might be required to break this stalemate.

The computationalist comes out of this debate relatively unscathed, having lost only the ability to have a knowably sound and complete competence theory that contains an account of our arithmetic activity. They are free to construct theories about other mental processes in this manner though, as Marr did with vision, so long as they omit talk of arithmetic. And as we saw, they can give accounts about our arithmetic activities under a formalist style of theory, so long as they are appropriately modest about their soundness claims. However, these are all only theoretical concerns at the moment: the actual process by which we do arithmetic is likely to be very complicated, so that we won't have the technological know-how to study it rigorously for some time. When we get there, it will most likely change the face of this debate. But until then, things remain at a stand still.

On the other hand, the Gödelian seems to have his hands tied. He has gone as far as formal reasoning can take him, but he has still not exceeded the capabilities of a machine. And upon reflection, this is not surprising. After all, formal logic is just a set of rules for the manipulation of symbols, and we know that machines are very good at applying such rules. Why then should we think that A) we can out do the machine in this way and B) demonstrate that we can do so? Thought about more generally, if one were able to show that he could out do any machine, then in order to convince others that this is the case they would need to communicate a proof to them. In order to do this though, they would have to represent whatever reasoning led them to this conclusion in a known symbolic language. But what is to stop a machine then from simulating this representation, thus "proving" it can out do any machine. In short, because our notion of "proof" implies the ability to communicate the truth of your statement, and it would be impossible to convey the truth of a mathematical statement without using symbols, it seems impossible to prove with mathematical certainty that you can outdo any given machine in a way that is inaccessible to the machine. It would require a very different notion of mathematical "truth" and "proof" before we could hope to give such a demonstration. Strangely enough, Gödel anticipates the eventual creation of such a paradigm, writing: "Namely, it turns out that in the systematic establishment of the axioms of mathematics, new axioms, which do not follow by formal logic from those previously established, again and again become evident. It is not at all excluded by the negative results mentioned earlier that nevertheless every clearly posed mathematical yes-or-no question is solvable in this way. For it is just this becoming evident of more and more new axioms on the basis of the meaning of the primitive notions that a machine cannot imitate"[11]. Put simply, the raw intuition we have that our unproven axioms are true is the type of reasoning that separates us from

machines. However, whether a systematic explication of this kind of intuition is possible is dubious, and at the very least a long way off. So it looks like things will remain locked up on the Gödelian side for the foreseeable future. And so we end our discussion, leaving the dialectic still in a draw, but with Penrose at least now squarely situated within it.

References

[1] P. Benacerraf, *God, the devil, and Gödel*, The Monist, vol. 51 (1967) pp. 9-32.

[2] M. Boden, *Computer Models of Mind*, Cambridge University Press, Cambridge, 1988.

[3] S. Bringsjord, K. Arkoudas, *The Modal Argument for Hypercomputing Minds*, Theoretical Computer Science, vol. 317 (2004), pp. 167-190.

[4] D.J. Chalmers, *Minds, machines, and mathematics: A review of Shadows of the mind, by Roger Penrose*, Psyche, vol. 2 issue 9, 1995.

[5] N. Chomsky, *Aspects of the Theory of Syntax*, MIT Press, Cambridge, MA, 1965.

[6] W. Craig, *On Axiomatizability within a system*, Journal of Symbolic Logic, Vol. 18, pp.30-32, 1953.

[7] S. Dehaene, E. Spelke, P. Pinel, R. Stanescu, S. Tsivkin, *Sources of Mathematical Thinking: Behavioral and Brain-Imaging Evidence*, Science, pp. 970-974, 1999.

[8] G. Gentzen, *Neue Fassung des Widerspruchsfreiheitsbeweises für die reine Zahlentheorie*, Forschungen zur Logik und zur Grundlegung der exakten Wissenschaften, vol. 4, pp. 19-44, 1938. Translated as 'New version of the consistency proof for elementary number theory', in (Gentzen, Szabo 1969).

[9] G. Gentzen, *Beweisbarkeit und Unbeweisbarkeit von Anfangsfällen der transfiniten Induktion in der reinen Zahlentheorie*, Mathematische Annalen, vol. 119, pp.140-161, 1943.

[10] K. Gödel, *Some basic theorems on the foundations of mathematics and their implications*, in [11], pp. 304-323, 1951.

[11] K. Gödel, *The modern development of the foundations of mathematics in the light of philosophy*, in [11], pp. 375-388, 1961.

[12] K. Gödel, *Collected Works, Vol III*, Oxford University Press, Oxford, 1995.

[13] D. Lewis, *Lucas against Mechanism II*, Canadian Journal of Philosophy, vol. 9, pp.373-376, 1979.

[14] P. Lindström, *Penrose's New Argument*, Journal of Philosophical Logic, vol. 30, pp.241-250, 2001.

[15] J.R. Lucas, *Minds, Machines, and Gödel*, Philosophy, vol. 36, pp. 112-137, 1961.

[16] J.R. Lucas, *Turn Over the Page*, talk for British Society for Philosophy of Science, 1996.

[17] D. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, MIT Press, Cambridge, MA, 1982.

[18] W. McCulloch, W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, Bulletin of Mathematical Biophysics, vol. 5, pp. 115-133, 1943.

[19] A. Newell, H. Simon, *Human problem solving*, Prentice-Hall, Englewood Cliffs, NJ, 1972.

[20] R. Penrose, *Shadows of the Mind: A Search for the Missing Science of Consciousness*, Oxford University Press, Oxford, 1994.

[21] R. Penrose, *The Emperor's New Mind: Concerning computers, minds, and the laws of physics*, Oxford University Press, Oxford, 1989.

[22] H. Putnam, *Minds and Machines*, Dimensions of Mind: A Symposium, New York University Press, New York, pp.138-164, 1960.

[23] Z. Pylyshyn, *Computation and Cognition: Toward a Foundation for Cognitive Science*, MIT Press, Cambridge, MA, 1984.

[24] S. Shapiro, *Mechanism, Truth, and Penrose's New Argument*, Journal of Philosophical Logic, vol. 32, pp.19-42, 2003.

[25] A. Turing, *On Computable Numbers, with an Application to the Entscheidungsproblem*, Proceedings of the London Mathematical Society, Vol. 42, 1937.